Scheduling supply chain node with fixed component arrivals and two partially flexible deliveries

Susana Carrera, Wahiba Ramdane-Cherif, Marie-Claude Portmann

Scheduling supply chain node with fixed components arrivals and two partially flexible deliveries

S. Carrera * **W. Ramdane-Cherif** * **M.C. Portmann** *

* *INPL-LORIA, ENSMN, Parc de Saurupt, CS 14234, 54042 Nancy Cedex France (e-mail: carrersu@loria.fr, portmann@loria.fr, ramdanec@loria.fr).*

Abstract: We consider here a logistic platform or more generally a node of a supply chain. After previous research works at the planning level whose aim was to smooth the workload by modifying slightly arrival and departure dates, we are now interested by the scheduling level. Our particular industrial framework led us to original hypotheses: given component quantities are delivered by trucks at some fixed times; a first optimized tour of the customers is planned at a known fixed date and a second optimized tour will be executed at a flexible date corresponding to the end of the schedule with the remaining customer orders. We reduce the activity inside the platform to the most important operation. This operation is performed by a single non renewable resource. Nevertheless most of the presented results could be easily extended to identical parallel machines. The considered scheduling problem is NP-Hard. With the goal of solving it by a branch and bound approach, we propose here a series of upper bounds (rapid approximation methods) and a series of lower bounds (obtained by various relaxations). Experimentations permit us to compare quality and computational times of the lower bounds and give us a first idea of the quality of the rapid approximation approaches.

1. INTRODUCTION

This research work has been developed after a logistic platform audit. The concerned firm is a shoes firm. The firm owns the platform, a vehicle fleet and about a hundred shops, but also rents complementary resources. This shoes firm completely manages the platform work and the downstream supply chain but not the upstream flows. Twice a year, spring and autumn, the platform receives products from its suppliers. A small part is kept in the reserve area to restock the chain stores, while the major part is immediately sent to the stores. The quantities and the delivery dates are negotiated with the suppliers at the planning level. The potential variations concerning delivery dates and quantities are strongly limited by the suppliers' production and storage constraints and by the transport organization between the suppliers and the platform. In Carrera et al. [2009] we tackled with the predictive planning for workload smoothing with seasonal demand in similar platform. We proposed integer linear programming models, as generic as possible, for smoothing the platform's workload, planning the workforce and negotiating quantities and dates for suppliers and clients deliveries. While at the planning level we considered the upstream flows can be slightly flexible, at the operational level the upstream flows are fixed and the arrival of products (different types of shoes) can be represented by given cumulated stairs curves, see Figure 1. On the other hand, the shoes distribution enterprise owns the stores and can manage completely the vehicle routing organization between the platform and its stores. In order to optimize the vehicle routing cost, optimized tours can be computed for subset of stores and dates and corresponding quantities of such

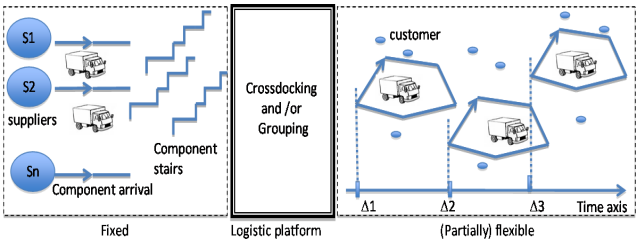

Fig. 1. Supply chain node

tours can be defined and slightly modified. Our scheduling problem concerns the preparation of shops orders that may contain different types of products. Products are shoes that can be differentiated by their category (man, woman, child), their colour and their size. We can distinguish two kinds of operations related to the preparation of the shop orders: first cross docking when products have to go through the warehouse directly without conditioning, and second Grouping when different products corresponding to a shop must be consolidated before delivery. The prepared shop orders must be delivered according to prefixed delivery dates that correspond to truck departures. The aim is to schedule the preparation of the shops orders with resource and delivery date constraints. We met other applications corresponding to the same scheduling model. We have unfortunately no place to present them here.
Corresponding scheduling literature is analyzed in section 2. In section 3 a specific scheduling problem is defined. Section 4 and 5 propose respectively rapid approximation methods and lower bounds. Computational evaluations are given in section 6. Finally section 7 contains a conclusion and some suggestions for further research.

2. STATE OF THE ART

Our problem is related to resource constrained scheduling literature and scheduling under delivery date constraints literature.

2.1 Resource constrained scheduling

In most of the scheduling problems involving resource constraints two types of resources are distinguished: renewable and non-renewable (or consumable) resources. Since the components in the supply chain node are considered as consumable, we will focus on this type of resources in our literature analysis. Blazewicz et al. [1986] identified two problems where the allocation of constrained resources has been mainly considered, these problems are: resource constrained project scheduling problem (RCPSP) and machine scheduling.
Carlier [1984] associated consumable resources to financial resources because of the similarity of the availability constraints they infer. He proves that RCPSP with financial constraints and arbitrary precedence constraints is polynomial if no renewable resource is considered. It becomes NP-hard if there are consumption and production of consumable resources. Patterson et al. [1989] proposed an exact procedure to solve the non preemptive project scheduling problem with precedence and resource constraints and with production and consumption activities. They also consider money as the single consumable resource. Carlier et al. [2009] dealt with the project scheduling problem where the units of resources are produced or consumed at the occurrence of precedence-related events. They proposed a list-scheduling based algorithm to minimize the makespan.
Carlier [1984] also provided a variety of complexity results on non preemptive one machine scheduling subject to financial constraints. The financial constrained one machine scheduling problem is NP-hard when the job processing times are not equal to one. Slowinski [1984] handled the preemptive job scheduling on parallel machines with financial constraints. The consumption rate of financial resource is constant during job processing. The author proposed a two-phase method based on linear programming to minimize the schedule length.
Cochand et al. [1989] took into account consumable resources with a time-varying supply (i.e. staircase and piecewise linear). They generalized the two-phase method to consider this type of resources. Toker et al. [1991] studied the case of one machine scheduling with a single financial resource which is continuously supplied at a constant rate. They proved that the problem is equivalent to a two-machine flowshop problem. Xie [1997] generalized the previous result to the single machine scheduling problem with multiple financial resource constraints, where the financial resources arrive uniformly over time.

2.2 Fixed delivery dates

Matsuo [1988] introduced an environment in which delivery dates are fixed and given a-priori before any jobs are processed. He proved that the total weighted tardiness and total tardiness problems with fixed shipping dates are NP-hard. In Hall et al. [2001], for a wide variety of objectives, the authors provided either a polynomial time algorithm or a proof of intractability of problems with fixed delivery dates.

This paper proposes to combine two scheduling areas: consumable resources and a generalization of the fixed delivery dates problems. To the best of our knowledge, this problem is new and has never been considered before.

3. PROBLEM DEFINITION

A set N of n independent jobs corresponding to the order preparation must be processed by the platform. Preemption of the jobs is assumed not to be allowed. One considered renewable resource can handle only one job at time. Each job is assumed to be available for processing when all necessary components are available and it consumes the used components. Each job j, $j \in N$, consumes $a^s_{j,i} \geq 0$ units of component i at the start (s) of its processing. The component represents in our case type of shoes (category, size, colour). They can arrive at different dates with fixed supplier deliveries. The arrival of each component can be represented as a cumulated stair curve. The processing time of job j is denoted by $p_j > 0$. In this paper, we consider a relatively short horizon in which only two truck tours are organized. The date of the first departure is fixed and equal to D. Any already prepared orders will be put in the trucks corresponding to this departure (no limit of capacity for this departure). D can be considered as a fixed common due date and any order not sent in the first tour will be considered as late. The order not yet completed at date D will be put in another tour, whose departure is flexible and equal to the global completion time $Cmax$. In consequence, we define $\hat{C}_j = D$ if $C_j \leq D$ and $\hat{C}_j = Cmax$ otherwise, where $Cmax$ is the completion time of the last job. When $\hat{C}_j = D$ the job j is on time and $\hat{U}_j = 0$. When $\hat{C}_j = Cmax$, the job j is late and $\hat{U}_j = 1$. A penalty ω_j is associated to each late job j and represents an estimation of loss of sales per unit time when orders arrive tardy to the shops. We search a schedule that minimizes the total tardiness cost defined as the global completion time multiplied by the sum of penalties associated to the tardy jobs. In order to eliminate the potential value 0 for our criterion, we add the value 1 to the sum of penalties. To the best of our knowledge, this scheduling problem was not yet considered. Using the well known $\alpha/\beta/\gamma$ notation, the problem could be denoted by $1/Stairs(nc), a^s_{j,i}, D/(1+\sum \omega_j \hat{U}_j) \times Cmax$, where $Stairs(nc)$ indicates nc consumable resources, whose cumulated arrival curves are stairs and D indicates one fixed delivery date for on time jobs.

4. RAPID APPROXIMATION METHODS

Our aim is to design rapid approximation methods, which will be included into branch and bound approaches in further research works. In consequence, we will use list algorithms based on priority orders to build active and/or no delay schedules. This section is divided into two parts. In the first one we recall how to build active and no delay schedules in the framework of the considered problem. In the second one we will propose a series of specific priority rules.

4.1 Schedule generators

Active schedule generator

Definition: A schedule is said to be active if no operation can be scheduled sooner without delaying at least another operation. In other words, no idle time is created, which can contain completely an operation scheduled later.

Generic algorithm for active schedule using a given order σ: $HA\sigma$
Let L be the job list sorted in σ order.
$t = 0$.
for each component i do
$CCCA_i$= cumulated curve of component i arrivals
$CCCU_i = [0, 0, \ldots, 0]$ = cumulated curve of component utilization
endfor
while L is not empty do
$MinC = +\infty$
for each job j do
place job j as soon as possible after time t so that the modified $CCCU$ curves remained smaller or equal to the $CCCA$ curves.
Let S_j and C_j be respectively the corresponding starting and completion time of job j.
$MinC = min(MinC, C_j)$.
endfor
let be j_0 the first job of list L such that $S_{j_0} < MinC$.
schedule j_0 between S_{j_0} and C_{j_0}.
modify the curves $CCCU$.
remove j_0 from list L.
$t = C_{j_o}$.
endwhile

No delay schedule generator:

Definition: A schedule is said to be no delay if no machine remains inactive while a job is waiting to be processed on this machine (here with the available components).

Generic algorithm for no delay schedule using a given order σ: $HND\sigma$
Let L be the job list sorted in σ order.
$t = 0$.
for each component i do
$CCCA_i$ = cumulated curve of component i arrivals
$CCCU_i = [0, 0, \ldots, 0]$ = cumulated curve of component utilization
endfor
while L is not empty do
$MinS = +\infty$.
No-idle-time = false.
for each job j of L until no-idle-time == true do
place job j as soon as possible after time t so that the modified $CCCU$ curves remained smaller or equal to the $CCCA$ curves.
Let S_j and C_j be respectively the corresponding starting and completion time of job j.
$MinS = min(MinS, S_j)$.
if $MinS == t$ then no-idle-time = true endif.
endfor
let be j_o the first job of list L such that $S_{j_0} == MinS$.
schedule j_o between S_{j_o} and C_{j_o}.
modify the curves $CCCU$.
remove j_o from list L.
$t = C_{j_o}$.
endwhile

4.2 Orders associated to the considered problem

To build orders or priority rules, three intuitive and heuristic techniques can be used which are antagonist:

- in order to keep the $CCCU$ curves under the $CCCA$ curves without delaying too much the jobs, it seems better to put first the longest jobs asking for the smallest quantities of components. This can be implemented by using for example the decreasing order of the processing times (LPT) divided by the Weighted Sum of the Components Demands ($WSCD$). The greater weights can correspond to components, which are known to arrive slightly later or could be identical. This order will be denoted by LPT-SCD (weights identical) or $LPT - WSCD$. It is to be noted that even for only one component, the order corresponding to decreasing values of processing time divided by the component demand does not minimize $Cmax$.
- in order to minimize the sum of tardiness penalties, it is better to put first the shortest jobs with the greatest penalties. This corresponds to the well know $WSPT$ order. Without taking into account the $CCCU$ and $CCCA$ curves, it is a good heuristic for the knapsack problem, which can be improved by choosing differently the two last jobs just before arriving at the D time, when the first trucks are leaving the platform.
- in order to minimize $Cmax$, minimizing the idle times seems a good idea, which leads to prefer no delay schedule, nevertheless a job scheduled sooner to minimize an idle time can ask too much components and create bigger idle times later.

We will use the previously defined orders either individually, or hierarchically (the second order is used only to break ties on the first one), or even dynamically. A dynamical use of orders consists in beginning with a given order and continuing with another one when some condition is verified such as $t \geq D$ or t greater than the last component arrival. In particular, $WSPT$ becomes useless when $t \geq D$ and $LPT - WSCD$ or $LPT - SCD$ becomes useless after the last component arrival. It is to be noted that if both $t \geq D$ and t greater than the last component arrival then any job order can be used to complete the schedule without any further control; it will be applied for any designed heuristic.

We get the following interesting orders:
$\sigma = 1 : LPT - WSCD/WSPT$,
$\sigma = 2 : WSPT/LPT - WSCD$,
$\sigma = 3 : DYN(LPT - WSCD \rightarrow WSPT)/WSPT$,
$\sigma = 4 : DYN(WSPT \rightarrow LPT - WSCD)/LPT - WSCD$

This provides us with at least 4 orders, which can be combined with the two presented generators in order to design 8 rapid approximation methods (more if we use various set of weights for the component demands in $WSCD$ and also if we improve the selection of the two last jobs just before D). As each method is rapid, we will apply the whole set of heuristics and call $BESTH$ the global heuristic consisting in applying successively all the methods and keeping the best solution obtained.

5. LOWER BOUNDS

5.1 Lower bounds using "agreeable orders"

Lower bound for Cmax

Lemma 1. When any job requires exactly the same number a_k of component for each component k, then LPT minimizes $Cmax$.

Proof. The proof is quite obvious by using a series of exchange pairwise on a schedule, which does not verify the order LPT. Under slightly different hypotheses: the component arrivals are uniform and continuous. This lemma was proposed by Xie [1997].

Lemma 2. If there is only one component o and the processing times of all jobs are identical (equal to p), then the increasing order CC of the component requirement minimizes $Cmax$.

Proof. The proof is identical to the proof of lemma 1 by using an exchange pairwise on a schedule, which does not verify the order CC.

Even if there is only one component o, for different processing times and component requirements, the increasing order of the component requirements divided by the processing times does not minimize the $Cmax$, in consequence we will use an usual technique called "agreeable orders" to get our lower bound. If there are several components, we will use the technique of "agreeable orders" independently for each component, this will provide us a lower bound for each component and we will keep the maximal value obtained as the lower bound associated to "agreeable orders".
Considering only one component o, let be Π the problem studied in this paper and let be $\Pi^{'}(o)$ a relaxation of the problem Π obtained as follows: the first job of $\Pi^{'}(o)$ gets the greatest processing time and the smallest component requirement, the second job of $\Pi^{'}(o)$ gets the greatest remaining processing time and the smallest remaining component requirements and so on until the last job. The jobs of $\Pi^{'}(o)$ are sorted simultaneously by decreasing values of processing times and increasing values of component requirements and the following lemma 3 can be applied.
Lemma 3. If the decreasing order of processing times LPT is identical to the increasing order of component requirement CC, then the orders are "agreeable" and minimize the makespan $Cmax$.

This lemma is a consequence of lemmas 1 and 2. The maximum of the $Cmax$ lower bound obtained for $Cmax$ by this technique for any component will be denoted by $LB|AGO|CT$ (lower bound with agreeable orders for last job completion time).

Lower bound for the weighted sum of tardiness

Lemma 4. For any interval $[0, H]$, the sum of the included idle times contained in any schedule built by using the agreeable order of lemma 3 is a lower bound of the sum of the included idle times for any feasible schedule on the interval $[0, H]$.

Lemma 4 is a consequence of lemma 3. Let be $UP/D = D - LB|AGO|IT(D)$ the difference between D (date of departure of the fixed delivery) and $LB|AGO|IT(D)$, the greatest lower bound of the idle times on $[0, D]$, i.e. an upper bound of the sum of processing times, which can be put on $[0, D]$ by taking into account the component arrivals.

To get an upper bound of the weighted sum of penalties of on time jobs, we must put on the interval $[0, UP/D]$ a subset of jobs, whose sum of processing times is smaller than UP/D and whose sum of penalties w_j is maximized. This problem is equivalent to a knapsack problem with only one constraint, in which the satisfaction values are the tardiness penalties, the weight values are the processing times and the capacity of the knapsack is equal to UP/D. To get an upper bound for this knapsack problem, we use the improved upper bound proposed by Martello et al. [1990].
The lower bound of the weighted sum of tardy jobs denoted by $LB|AGOK|ST$ (lower bound for sum of tardiness penalties using agreeable orders and Knapsack relaxations) is obtained by subtracting the obtained upper bound from the total sum of tardiness penalties.

5.2 Lower bounds using integer relaxations

Relaxation using preemption and continuous consumption of components for Cmax

A relaxation can be obtained by accepting job preemption and by assuming that the components are not required at the start of the jobs, but are uniformly and continuously consumed during the job execution. The obtained problem is a particular case of the problem considered by Cochand et al. [1989], which use linear programming to solve it. The authors used a two-phase procedure, but only the first phase is used here because getting a feasible solution for their problem is useless for computing our lower bound.
Assuming the time axis is divided into periods $1, 2, \ldots, h$ where h is the number of resource arrivals having durations $T^1, T^2, \ldots, T^h$. A linear program determines the time that each job j is processed in each sub-period T^h while minimizing the makespan and respecting the component arrival curves.
The lower bound for $Cmax$ obtained by applying the method of Cochand et al. [1989] on this relaxed problem will be denoted by $LB|IR|CT$ (lower bound using integer relaxation for last job completion time).

Relaxation using preemption and continuous consumption of components for the weighted sum of tardiness

Using the same relaxation as for $Cmax$, we can adapt the linear program in order to get an upper bound of the sum of on time job penalties. We have only to cut the horizon at time D and to modify the objective function, which consists now in maximizing the penalties associated to the proportion of jobs scheduled before time D. The constraints on the component arrivals are obviously kept. As previously, this upper bound for on time jobs is transformed in lower bound for late jobs. The lower bound obtained with this relaxation is denoted by $LB|IR|ST$

(lower bound using integer relaxation for the sum of tardy job penalties).

5.3 Lower bound using uniform and continuous component arrivals for Cmax

If there are several components, the same relaxation will be executed for any component. To simplify the explanations, we consider arbitrarily one of the components. Assume that the component level at time t is equal to CCA_t. We relax the staircase component arrival curve by taking a linear curve, which begins at the point (t, CCA_t). This linear curve is tangent at some points (at least one), but always greater or equal to the staircase component arrival curve, see Figure 2.

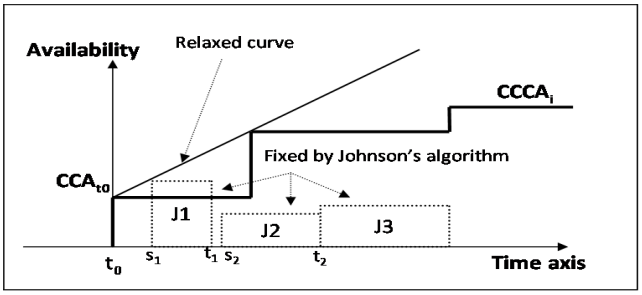

Fig. 2. TKEX relaxation for a resource

The problem with continuous supply of resource has been polynomially solved by Toker et al. [1991] when there exists only one component and by Xie [1997] for several components. The authors demonstrated that the problem $1/Cont(1), a_j/Cmax$ corresponds to a two machine flowshop without resource constraints where the processing times in machines one and two $p_{j,1}$ and $p_{j,2}$ are a_j and p_j respectively. Idle times in the original problem correspond to idle times in the second machine in the flowshop problem. Thus, $1/Cont(1), a_j/Cmax$ can be solved using Johnson's rule for the two-machine flowshop problem. As this relaxation could be too large particularly at the beginning of the schedule, we improve it by an iterative procedure.

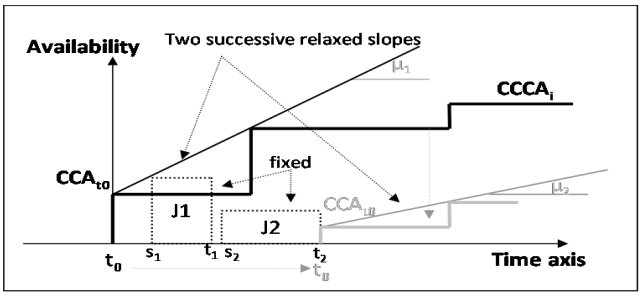

Fig. 3. Successive TKEX relaxation for a resource

We consider the schedule obtained by Johnson algorithm and we observe successively the end of each job beginning by the first job of Johnson's schedule. At each end of job, we examine the relaxed problem in which the beginning of the schedule is fixed and given by the used relaxation, but the end of the schedule is free and can be relaxed using again the same relaxation for the remaining sub problem. If for this new relaxation, the slope of the relaxed continuous arrival is strictly smaller than the previous slope for the considered component, then we solve the remaining sub problem with the new slope. The considered improvement can be again applied to the sub problem itself, see Figure 3.

The lower bound for $Cmax$ obtained with this relaxation will be denoted by $LB|UCA|CT$ (lower bound with uniform and continuous arrival relaxation for last job completion time). The lower bound of the global considered problem will be obtained by the formula:

$LB = (LBmaxST + 1) \times LBmaxCT$, where $LBmaxCT = max(LB|AGO|CT, LB|IR|CT, LBT|UCA|CT)$ and $LBmaxST = max(LB|AGOK|ST, LB|IR|ST)$

6. COMPUTATIONAL EXPERIENCE

The experimental section first compares the performances of the three $Cmax$ lower bounds. The performances of the various upper bounds provided by the heuristic approaches are also evaluated by comparison with the best lower bound values. Tests have been computed for four sets of instances. Two parameters are used to differentiate families of generated instances: dispersion of the component arrival dates and position of the on time delivery date D. The horizon length is estimated by using one of the upper bound of the $Cmax$ value. Component arrivals can be dispersed over the time horizon (denoted by DA) or relatively grouped at the beginning of the horizon (denoted by RA). The on time delivery date is generated either at the middle of the horizon (denoted by MD) and at the third quarter of the horizon (denoted by GD). This provides us with four sets of instances denoted by RA/GD, RA/MD, DA/GD and DA/MD. Each set of instances contains 20 instances with $n = 10$ or 20 or 50 jobs. The three $Cmax$ lower bounds are compared in Table 1. For each lower bound and data set, the average computation time (CPU) is given in seconds. The performance is given by column GAP containing the average error percentage (difference between the best lower bound and the current lower bound divided by the current lower bound and multiplied by 100).

Table 1. Makespan lower bound comparison

Data Set	$LB\|AGO$		$LB\|IR$		$LB\|UCA$	
	GAP	**CPU**	**GAP**	**CPU**	**GAP**	**CPU**
RA/GD	0.52	0.00	0	0.08	0.65	0.02
RA/MD	0.16	0.00	0.14	0.04	0.19	0.03
DA/GD	5.79	0.02	0	0.11	10.4	0.03
DA/MD	4.89	0.01	0	0.07	13.3	0.03

In Table 1, for all instance sets, we can see that the best $Cmax$ lower bound is $LB|IR$ followed by $LB|AGO$ and finally $LB|UCA$.

The comparison between the two lower bounds for the sum of weighted tardiness is presented in Table 2.

Table 2. Weighted Tardiness lower bound comparison

Data Set	$LB\|AGOK$		$LB\|IR$	
	GAP	**CPU**	**GAP**	**CPU**
RA/GD	1.68	0.03	2.08	0.05
RA/MD	0.31	0.04	0.86	0.05
DA/GD	30.1	0.03	0.03	0.04
DA/MD	15.5	0.03	0.07	0.05

$LB|AGOK$ is better for regrouped arrivals, while $LB|IR$ is considerably better for dispersed arrivals.
In Table 3 we compare the approximation approaches with the lower bounds. The column GAP corresponds again to the average error percentages between the best upper bounds and the best lower bounds associated to each instance. The performances are given separately for three criteria: $Cmax$ (total duration only), $1 + \sum \omega_i \hat{U}_i$ (ST: sum of tardiness penalties) and $(1 + \sum \omega_i \hat{U}_i) \times Cmax$ (GC: Global considered Criterion). Furthermore the column BEST H contains the percentage of times a rapid approximation approach provides the best found value for the global criterion. Only the name of the three best heuristic (HNDσ or HAσ, as defined in section 4) is given with the corresponding percentage.

Table 3. Upper and lower bounds comparison

Data Set	GAP			BEST H	
	Cmax	**ST**	**GC**	H	%BEST
RA/GD	0.61	15.2	16.1	HND_2	27.9
				HND_4	27.9
				HND_3	14.2
RA/MD	1.01	9.18	10.9	HND_2	32.5
				HND_4	32.5
				HA_2	17.5
DA/GD	4.59	31.8	39.5	HA_1	15
				HA_3	15
				HND_3	13.8
DA/MD	2.82	14.2	20.6	HA_4	22.5
				HND_2	20
				HND_4	20

The evaluation of the performances of the approximation approaches compared with the lower bounds shows that the GAPs for $Cmax$ is quite good for families RA/GD and RA/MD and less good for DA/GD and DA/MD. One explanation is that any idle time due to resource constraints disappears after the last component arrival and the potential error of the approximation approaches are only made for the subset of jobs placed on the first half of the horizon (with RA instances). For the weighted tardiness, even by taking into account the resource constraints, using an upper bound of the idle times on the interval $[0, D]$, the GAPs remain quite large, which induces naturally large GAPs for the global criterion. Two ways can decrease them, either to design even better lower bounds or to improve the approximation methods. The later is certainly easier.

7. CONCLUSIONS AND FURTHER WORKS

We propose in this paper a new scheduling problem arising in the framework of supply chain nodes such as logistic platforms. It crosses two families of known but not excessively studied scheduling problems, families with consumable resource arrivals and families with fixed dates for deliveries. Moreover we consider only two deliveries, the first delivery date is fixed, but the second one is flexible and associated to the end of the last job. Tardiness penalties are associated to the late jobs delivered with the second delivery and the tardiness cost is proportional to the duration of the lateness. We have developed lower bounds and upper bounds for any potential criterion. These bounds are evaluated by appropriate experiments. We present here the results with a single machine as renewable resources. We have implemented a branch and bound method using the results of this paper, a well known separation scheme and some dominance properties. Promising results were obtained by preliminary tests. This encourage us to continue to improve the branch and bound method.
The results of this paper can be very easily extended to identical parallel machines by considering a global equivalent machine for the lower bounds and by placing the jobs on the first available machine for the upper bounds. Other perspectives consist in changing the criteria, considering for example only the total duration (makespan) for which there are already some results in the literature (for relaxed problems providing us our lower bounds) or considering several fixed delivery dates and one of the usual tardiness criteria. Finally, both exact and approximation methods such as meta heuristic can be developed.

REFERENCES

J. Blazewicz, W. Cellary, R. Slowinski, and J. Weglarz. Scheduling under resource constraints – deterministic models. *Annals of Operations Research*, Vol 7, J.C. Baltzer AG, Basel, 1986.

J. Carlier. Problèmes d'ordonnancements à contraintes de ressources: Algorithmes et complexité. *Doctorat d'état*, 1984.

J. Carlier, A. Moukrim, and H.Xu. The project scheduling problem with production and consumption of resources: A list-scheduling based algorithm. *Discrete Applied Mathematics*, 157(17):3631–3642, 2009.

S. Carrera, M.C. Portmann, and W. Ramdane Cherif. Outils d'aide à la décision pour le lissage de charges des plateformes logistiques. *Congrès International de Génie Industriel 2009*. 8p. 10-12 juin, Tarbes-France.

M. Cochand, D. de Werra, and R. Slowinski. Preemptive scheduling with staircase and piecewise linear resource availability. *Methods and Models of Operations Research*, 33:297–313, 1989.

N.G. Hall, M. Lesaoana, and C.N. Potts. Scheduling with fixed delivery dates. *Operations Research*, 49(1):134–144, 2001.

S. Martello and P. Toth. *Knapsack Problems*. John Wiley & Sons, New York, 1990.

H. Matsuo. The weighted total tardiness problem with fixed shipping times and overtime utilization. *Operations Research*, 36(2):293–307, 1988.

J.H. Patterson, R. Slowinski, F.B. Talbot, and J. Weglarz. An algorithm for a general class of precedence and resource constrained scheduling problems. In R. Slowinski and J. Weglarz, editor, *Advances in Project Scheduling*, pages 3–28. Elsevier Science Publishers B.V., Amsterdam, 1989.

R. Slowinski. Preemptive scheduling of independent jobs on parallel machines subject to financial constraints. *European Journal of Operational Research*, 15:366–373, 1984.

A. Toker, S. Kondakci, and N. Erkip. Scheduling under a non-renewable resource constraint. *Journal of the Operational Research Society*, 42(9):811–814, 1991.

J. Xie. Polynomial algorithms for a single machine scheduling problems with financial constraints. *Operations Research Letters*, 21:39–42, 1997.